Lezione n.9
Peer-to-Peer Systems and Applications
Capitolo 8

PASTRY

Laura Ricci

PASTRY

- Pastry: proposto nel 2001 da Rowstron (Rice University) e Druschel (Microsoft)

- Obiettivo principale: definire un middleware per la costruzione di applicazioni P2P di tipo diverso

  File sharing

  Memoria distribuita

  Group Communication

  ….

- Caratteristiche:

  decentralizzazione completa

  routing efficiente

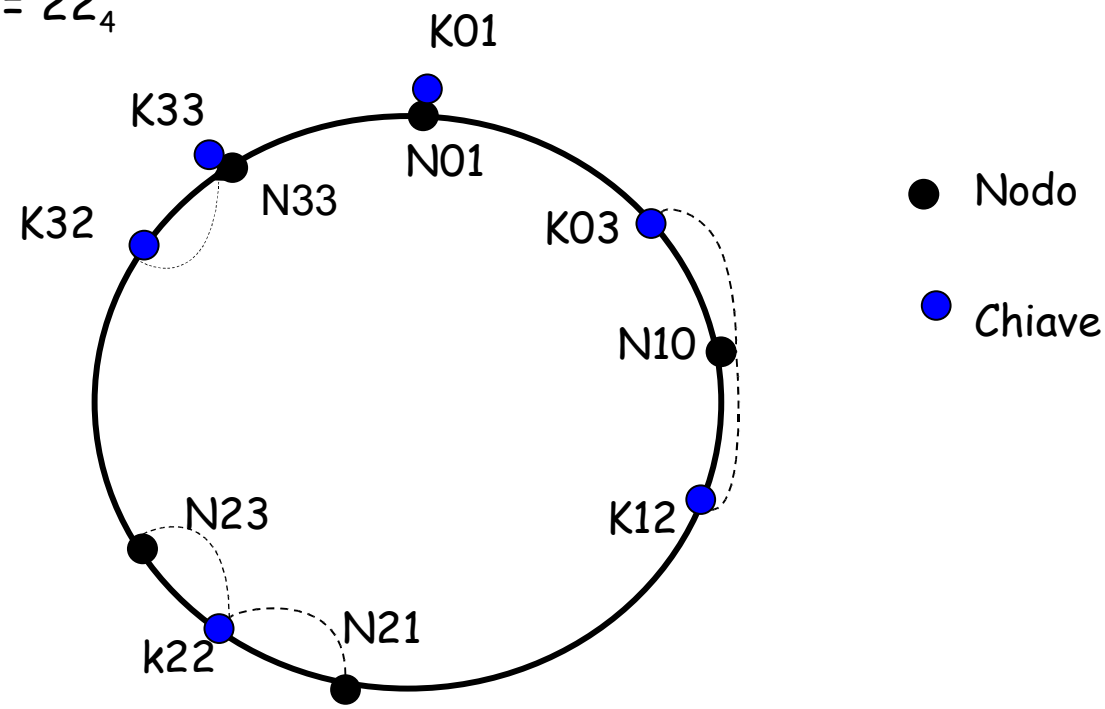
PASTRY

- Pastry associa ad ogni nodo (nodeID) e ad ogni chiave (key) un identificatore, mediante una funzione hash
- Identificatori di l bits (l in genere uguale a 128)
- Le sequenze di bits vengono interpretate come valori in base 2^b (in genere b=4) (Es: 100011111010=8FA, per b=4)
- Ogni chiave viene associata al nodo il cui nodeID ha valore più vicino al valore della chiave, tra i nodi attivi sulla rete Pastry
- Nel caso in cui vi siano più nodi a uguale distanza numerica dalla chiave, la chiave viene replicata su ognuno di essi

PASTRY

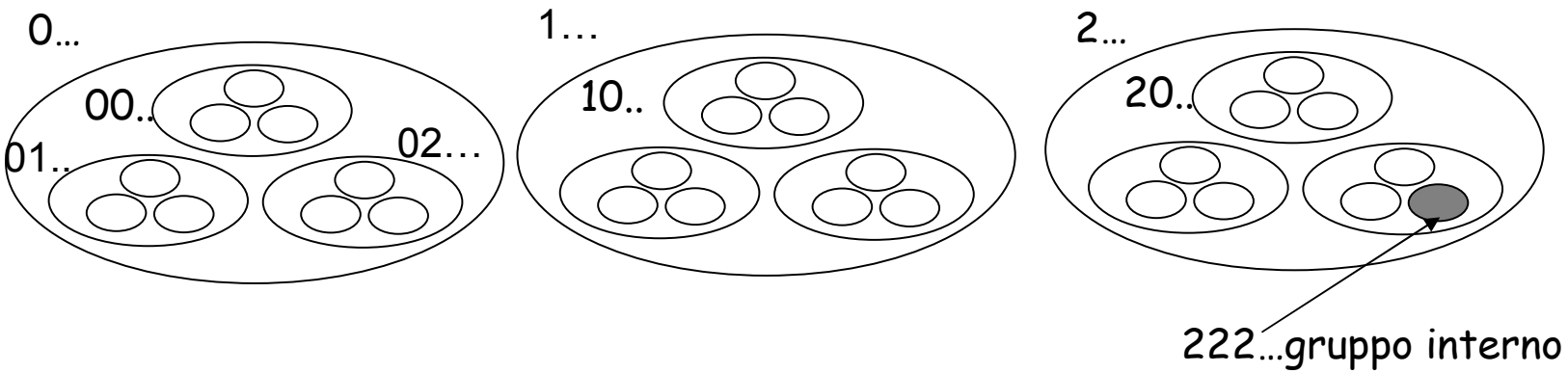
- Spazio degli identificatori con l=4, 4 bits, b=2. Ogni stringa di 4 bits viene interpretata come un valore in base 2^b=4

- Esempio: $1010_2 = 22_4$



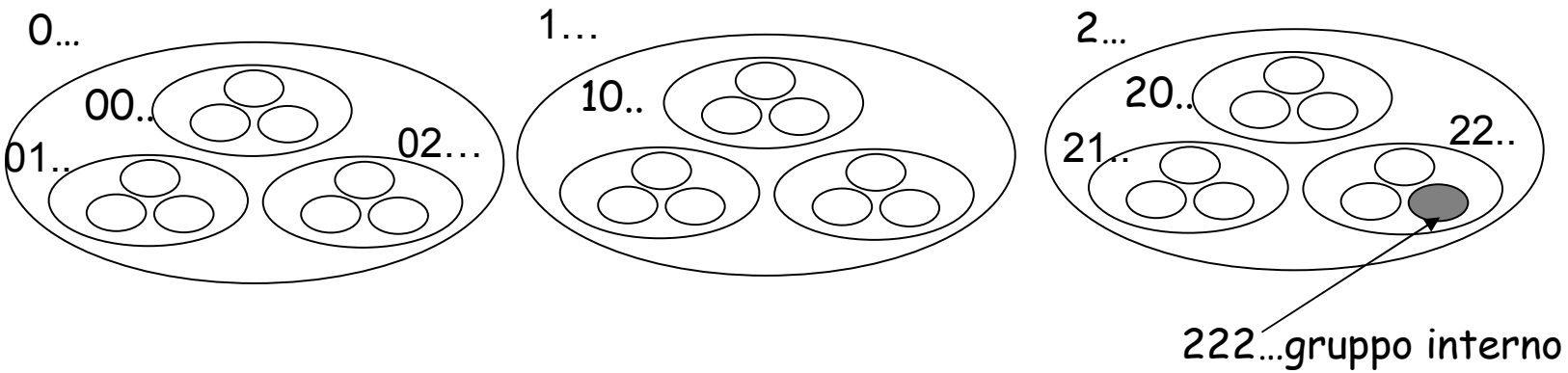
Pseudo Pastry: Concetti Base

- Esempio: alle chiavi ed ai nodi sono associati identificatori di n cifre in base es: 02112100101022
- Ogni chiave K viene memorizzata nel nodo con il valore dell'id più vicino al valore di K
- I nodi sono logicamente raggruppati in base al loro indirizzo



Pseudo Pastry: Concetti Base

- Ogni nodo appartenente ad uno dei gruppi più interni conosce l'indirizzo IP di ogni altro nodo dello stesso gruppo
- Ogni nodo conosce l'indirizzo IP di un nodo "rappresentante" di ogni altro gruppo
- Nodo 222….: conosce 0…,1…,20…,21…,220…,221…
- In questo modo il nodo 222 mantiene informazioni su 6 "rappresentanti", invece di 27

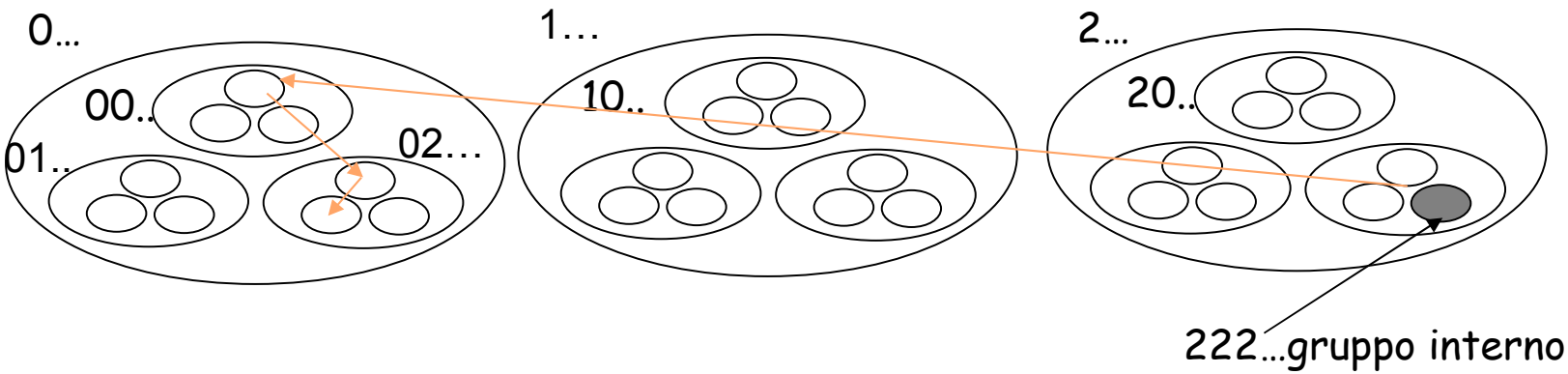


Pseudo Pastry: Concetti Base

- Supponiamo che un nodo del gruppo 222… voglia ricercare la chiave k=02112100210. Il nodo adotta una strategia di tipo divide and conquer

- K viene inoltrata ad un nodo "rappresentante" del gruppo 0…, poi al nodo "rappresentante" di 02,…. quindi a quello di 021

- 021…. inoltra la chiave al nodo più vicino alla chiave numericamente



Pastry: Tabelle di Routing

- Ai nodi vengono assegnati identificatori di 128 bits

- Ogni identificatore in base $2^4 = 16$

  Esempio 65a1fc04

  Ogni gruppo contiene 16 sottogruppi

- Ogni nodo gestisce una tabella di routing ed un leaf set

  La tabella di routing contiene un riferimento ad un nodo delegato per ogni gruppo

  Il leaf set contiene riferimenti agli altri nodi del gruppo interno

Pastry: Tabella di Routing

Tabella di routing per il nodo n=65a1fc04

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		**65bx**	**65cx**	**65dx**	**65ex**	**65fx**
						.....	.....	.....	.....						

- $\log_{16}$ N righe
- 2^b-1 entrate per ogni riga
- x = stringa di cifre esadecimali

Sul nodo n, gli elementi della riga i della tabella contengono riferimenti ai nodi il cui NodeID

- condivide un prefisso di lunghezza i con il NodeID di n
- differisce nella i-esima cifra rispetto al NodeID di n

Pastry: Tabella di Routing

- Alcune entrate della tabella di Routing possono essere vuote

- Approccio analogo a Chord. Ogni tabella di routing contiene una conoscenza:
  - approssimata rispetto ai nodi distanti nello spazio degli identificatori
  - più accurata dei nodi vicini

- Proximity Routing
  - la scelta dei nodi da inserire nella routing table di un nodo è guidata da un criterio di prossimità
    - ping delay
    - numero di IP hops
  - ogni elemento della tabella di routing di n si riferisce ad un nodo vicino ad n (con riferimento alla nozione di prossimità utilizzata), tra tutti i nodi con il prefisso appropriato per quella entrata

Pastry: l'algoritmo di Routing

Routing effettuato sul nodo n di identificatore ID

if (KEY appartiene all'intervallo definito da *leaf set*)

inviare key al nodo del leaf set il cui NodeID è numericamente più vicino a KEY

else

{

if (esiste nella tabella di routing un identificatore ID' che condivide con la chiave un prefisso più lungo del prefisso comune tra ID e KEY)

invia KEY al nodo identificato da ID'

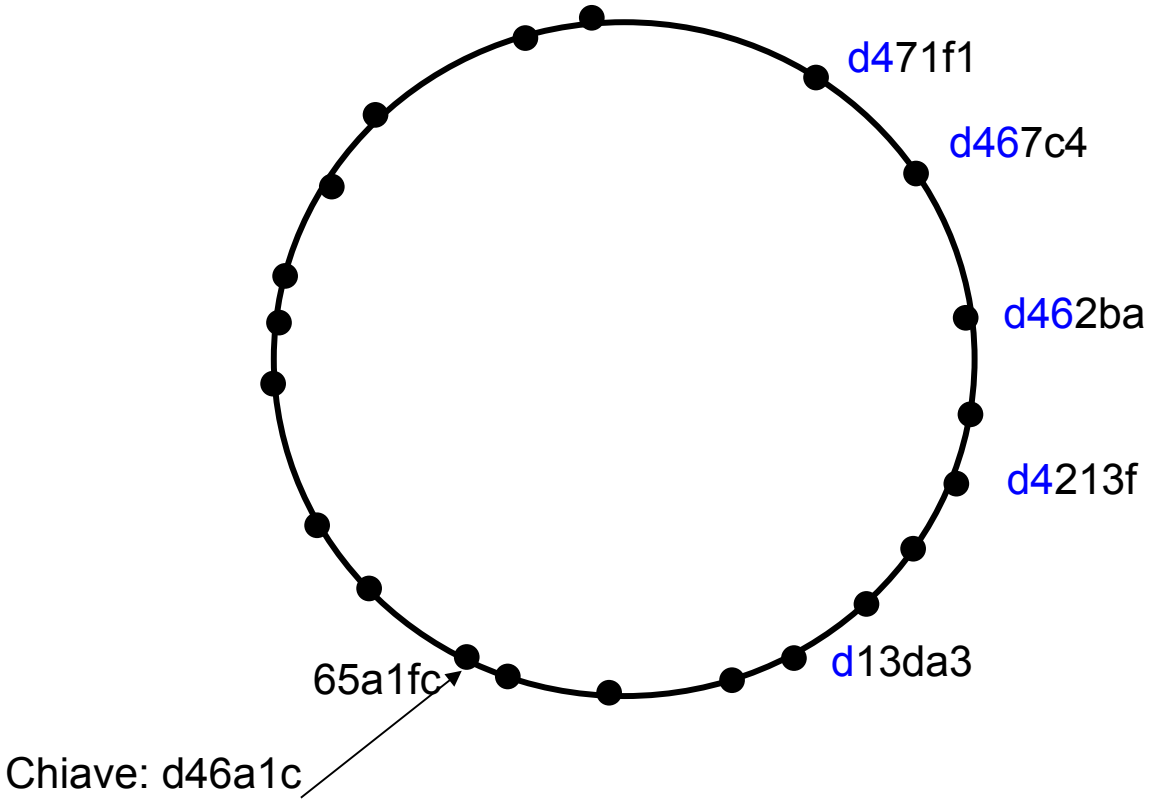
else

invia KEY ad un nodo che

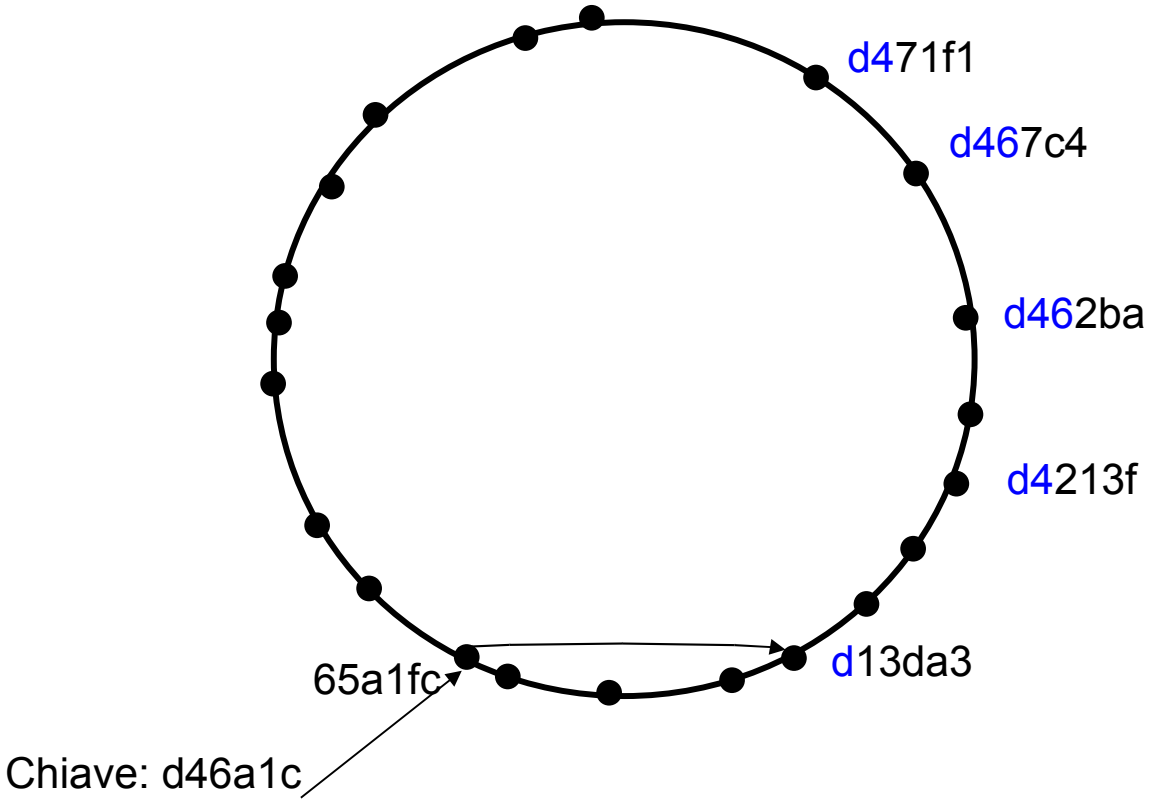
condivida con la chiave un prefisso di lunghezza pari al prefisso comune tra ID e KEY e sia numericamente più vicino alla chiave

}

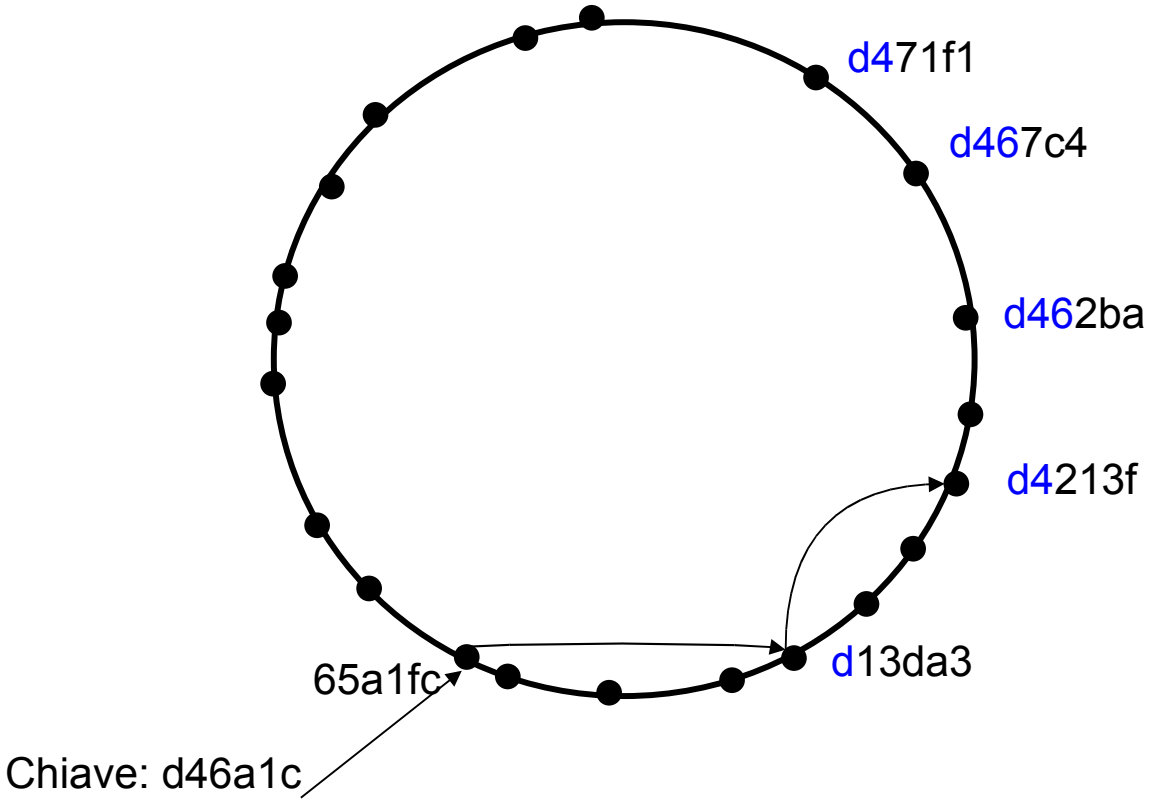
Pastry: l'Algoritmo di Routing



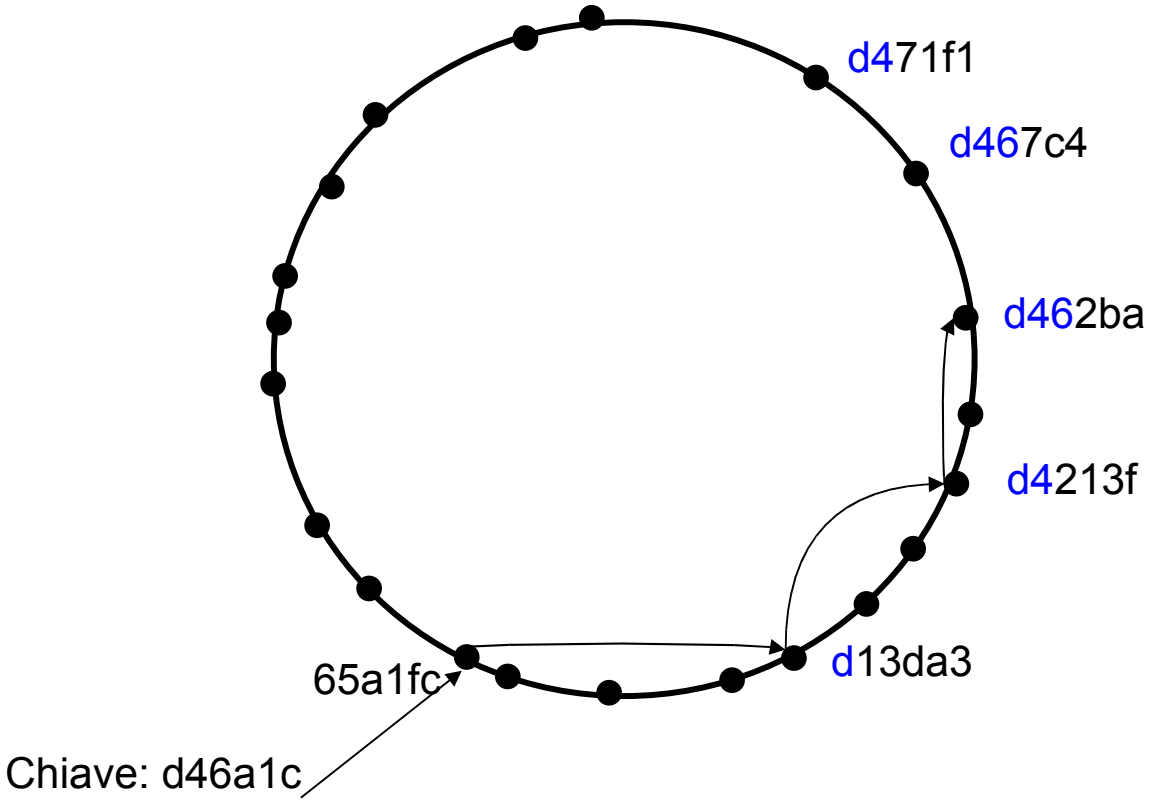
Pastry: l'Algoritmo di Routing



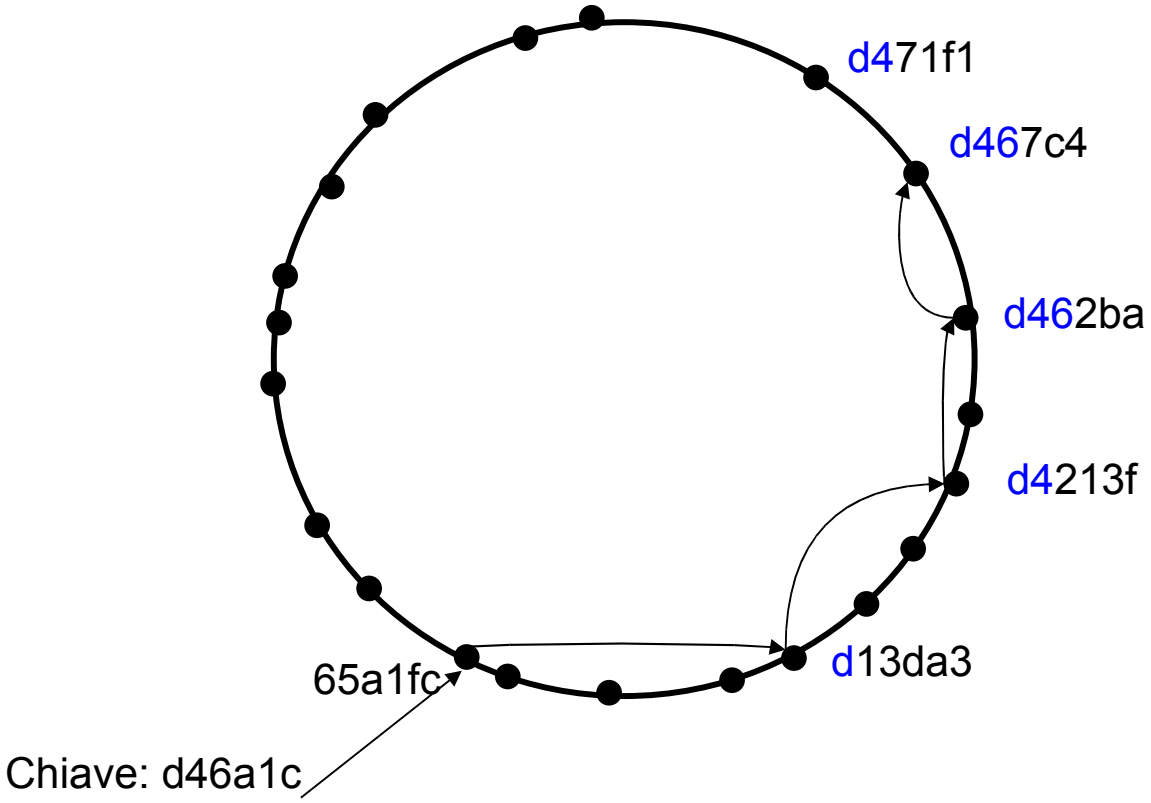
Pastry: l'Algoritmo di Routing



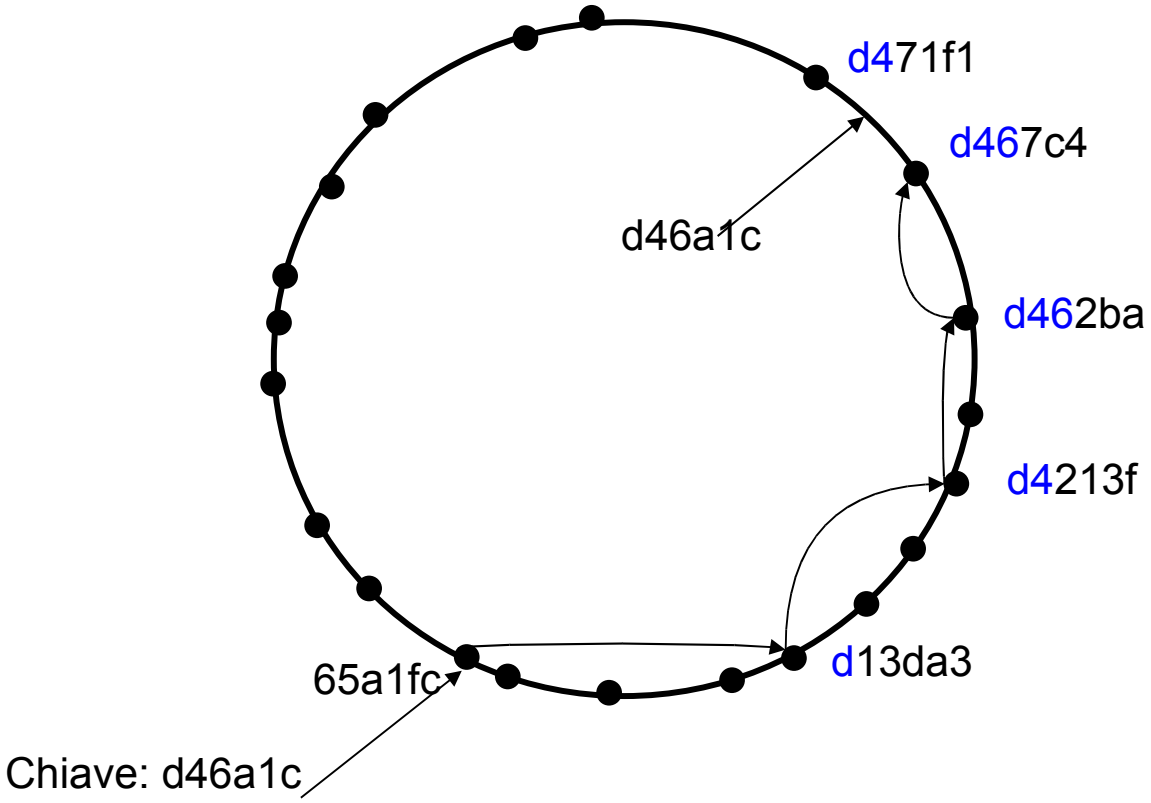
Pastry: l'Algoritmo di Routing



Pastry: l'Algoritmo di Routing



Pastry: l'Algoritmo di Routing



Pastry: L'algoritmo di Routing

Tabella di routing per il nodo n=65a1fc04

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
650x	**651x**	**652x**	**653x**	**654x**	**655x**	**656x**	**657x**	**658x**	**659x**		65bx	**65cx**	**65dx**	**65ex**	**65fx**
						.....	.....								

n riceve una richiesta per la chiave k=65b23c05

- 65 = prefisso comune tra il nodo e la chiave
- n invia la chiave al nodo evidenziato della tabella, perché questo nodo condivide un prefisso più lungo con la chiave (65b)

Pastry: L'algoritmo di Routing

Tabella di routing per il nodo n=65a1fc04

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
	651x	**652x**	**653x**	**654x**			**657x**	**658x**	**659x**		**65bx**	**65cx**	**65dx**		**65fx**
						.....									

Ipotesi: la parte non evidenziata della tabella è vuota

n riceve una richiesta per la chiave 65523c05

- 65 = prefisso comune tra nodo e chiave
- non esiste un riferimento ad un nodo che condivide un prefisso più lungo con la chiave,
- la chiave viene inviata al nodo evidenziato in tabella, perché il suo valore numerico è più vicino alla chiave

Pastry: Inserimento di Nuovi Nodi

Quando un nodo vuole inserirsi in una rete Pastry

- sceglie un identificatore *ID*, mediante *consistent hashing*
- sceglie un peer PB per il bootstrap
- invia a PB un messaggio *join(key=ID)*
- il messaggio viene inoltrato verso il nodo C numericamente più vicino ad ID
- la tabella di routing del nodo viene costruita a partire dalle tabelle di routing di tutti i nodi sul cammino percorso dal messaggio *join(key=ID)*
- ID notifica la propria presenza a questi nodi, che modificano le proprie tabelle di routing

Pastry: L'algoritmo di Routing

Tabella di routing per il nodo n=65a1fc04

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
	651x	**652x**	**653x**	**6543x**			**657x**	**658x**	**659x**		**65bx**	**65cx**	**65dx**		**65fx**
						.....									

Ipotesi: la parte non evidenziata della tabella è vuota

n riceve un messaggio join(key=65b2ff03) dal nuovo nodo nn= 65b2ff03

- le prime due righe della tabella di routing di n possono essere utilizzate per inizializzare le prime due righe della tabella di routing di n
- il messaggio viene inoltrato al nodo evidenziato in tabella

PASTRY: Complessità

Analisi della complessità

- Rete Pastry di N nodi. Identificatori di l-bits, valori in base $B=2^{b,}$
- Complessità ricerca chiavi: $O(\log_B (N))$
- Complessità tabelle di routing: $O(\log_B(N))x(B-1))$
- Valore di B scelto come tradeoff tra numero medio di passi per la ricerca e dimensione delle tabelle di routing (in generale b=4)

Pastry: Località

- Pastry non ottimizza solamente il numero di passi di routing, ma anche il costo di ogni hop
- La riga i-esima della tabella di routing contiene riferimenti ai nodi che condividono un prefisso lungo i con il nodo.
- Più scelte possibili per ogni elemento per la tabella
- Ogni elemento della tabella di routing di n contiene un riferimento ad un nodo vicino ad n (secondo una nozione di prossimità fisica), scelto tra tutti i nodi caratterizzati dal prefisso opportuno
- Prossimità fisica: ping delay, numero di hops IP